Phạm Trọng Đại - 18020264

**Bài toán tính tiền trà sữa**

* Input: loại trà sữa, số lượng
* Output: giá tiền phải trả

Điều kiện: Có 2 loại trà sữa:

* Trân châu đường đen 30k (1)
* Trân châu đường hổ 40k (2)

Đặt 2 cốc trở lên giảm 25%, giảm 50% khi đặt 4 trở lên, đặt tối đa 10 cốc.

**KIỂM THỬ HỘP TRẮNG**

**CODE**

import unittest

dict\_price = {

1: 30000,

2: 40000

}

def count\_money(type, number):

price = dict\_price[type]

if number < 2:

return price

elif number >= 2:

return price \* number \* 0.75

elif number >= 4:

return price \* number \* 0.5

class TestStringMethods(unittest.TestCase):

def test\_whitebox\_1(self):

self.assertEqual(count\_money(1, 1), 30000)

def test\_whitebox\_2(self):

self.assertEqual(count\_money(1, 2), 45000)

def test\_whitebox\_3(self):

self.assertEqual(count\_money(1, 4), 60000)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main(verbosity=2)

**Đồ thị dòng điều khiển cho hàm def count\_money(type, number)**

**Diagram

Description automatically generated**

**Test path, test case (độ phủ C2):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test path** | **Test case** | **Expected Output** | **Actual Output** | **Status** |
| 1 | 1,2(T),5 | type = 1, number = 1 | 30000 | 30000 | True |
| 2 | 1,2(F),3(T),6 | type = 1, number = 2 | 45000 | 45000 | True |
| 3 | 1, 2(F),3(F),4(T),7 | type = 1, number = 4 | 60000 | 90000 | False |
| 4 | 1, 2(F),3(F),4(F) | Không có | - | - | - |

**Nhận xét:**

Case 3 sai vì đặt câu lệnh if thứ tự không đúng.

Phải check >= 4 trước >= 2.

**KIỂM THỬ HỘP ĐEN**

**I - Kiểm thử giá trị biên xấu nhất**

+ Biến type: giá trị 1 hoặc 2

+ Biến number: 1<= number <= 10, là số nguyên

Min number: 1

Max number: 10

Min+ number: 2

Max- number: 9

Nom number: 5

**Test case:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TestID | Input | | Output | | Test status |
| Đồ uống | Số lượng | Expected | Actual |
| 1 | 1 | 1 | 30000 | 30000 | True |
| 2 | 1 | 2 | 45000 | 45000 | True |
| 3 | 1 | 5 | 75000 | 112500 | False |
| 4 | 1 | 9 | 135000 | 202500 | False |
| 5 | 1 | 10 | 150000 | 225000 | False |
| 6 | 2 | 1 | 40000 | 40000 | True |
| 7 | 2 | 2 | 60000 | 60000 | True |
| 8 | 2 | 5 | 100000 | 150000 | False |
| 9 | 2 | 9 | 180000 | 270000 | False |
| 10 | 2 | 10 | 200000 | 300000 | False |

**Phân tích kết quả kiểm thử:**

Case 3, 4, 5, 8, 9, 10 sai vì đặt câu lệnh if thứ tự không đúng.

Phải check >= 4 trước >= 2.

**II – Kiểm thử bảng quyết định**

**Bảng quyết định:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | R1 | R2 | R3 | R4 | R5 | R6 |
| Điều kiện | type = 1? | T | T | T | F | F | F |
| type = 2? | - | - | - | T | T | T |
| number <2 ? | T | F | F | T | F | F |
| number >= 2 ? | - | T | T | - | T | T |
| number >= 4? | - | - | T | - | - | T |
| Hành động | Giảm 25% | F | T | F | F | T | F |
| Giảm 50% | F | F | T | F | F | T |
| Không giảm | T | F | F | T | F | F |

**Test case:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TestID | Input | | Output | | Test status |
| type | number | Expected | Actual |
| R1 | 1 | 1 | 30000 | 30000 | True |
| R2 | 1 | 2 | 45000 | 45000 | True |
| R3 | 1 | 5 | 75000 | 112500 | False |
| R4 | 2 | 1 | 40000 | 40000 | True |
| R5 | 2 | 2 | 60000 | 60000 | True |
| R6 | 2 | 5 | 75000 | 112500 | False |

**Phân tích kết quả kiểm thử:**

Case 3, 4, 5, 8, 9, 10 sai vì đặt câu lệnh if thứ tự không đúng.

Phải check >= 4 trước >= 2.

**CODE**

def test1(self):

self.assertEqual(count\_money(1, 1), 30000)

def test2(self):

self.assertEqual(count\_money(1, 2), 45000)

def test3(self):

self.assertEqual(count\_money(1, 5), 75000)

def test4(self):

self.assertEqual(count\_money(1, 9), 135000)

def test5(self):

self.assertEqual(count\_money(1, 10), 150000)

def test6(self):

self.assertEqual(count\_money(2, 1), 40000)

def test7(self):

self.assertEqual(count\_money(2, 2), 60000)

def test8(self):

self.assertEqual(count\_money(2, 5), 100000)

def test9(self):

self.assertEqual(count\_money(2, 9), 180000)

def test10(self):

self.assertEqual(count\_money(2, 10), 200000)

def test\_r1(self):

self.assertEqual(count\_money(1, 1), 30000)

def test\_r2(self):

self.assertEqual(count\_money(1, 2), 45000)

def test\_r3(self):

self.assertEqual(count\_money(1, 5), 75000)

def test\_r4(self):

self.assertEqual(count\_money(2, 1), 40000)

def test\_r5(self):

self.assertEqual(count\_money(2, 2), 60000)

def test\_r6(self):

self.assertEqual(count\_money(2, 5), 100000)